|  |  |
| --- | --- |
|  |  |
| Operating System Project |  |
|  |  |
|  | Mostafa Medhat MohamedMohamed Essam SayedYusuf Ahmed Yusuf |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| How to add your system call the Linux OS kernel. |
|  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | System Descriptions:Ram: 6 GBProcessors: 4 coresHard Disk: 30 GBKernel Version: 5.8.0-55-genericSteps of How Adding the system call: **Section 1 – Preparation**  *In this section, you will download all necessary tools to add a basic system call to the Linux kernel and run it. This is the only part of the entire process where network connectivity is necessary.*   * 1. **- Fully update your operating system.**   *sudo apt update && sudo apt upgrade -y*       * 1. **- Download and install the essential packages to compile kernels.**   *sudo apt install build-essential libncurses-dev libssl-dev libelf-dev bison flex -y*     * 1. **- Clean up your installed packages.**   *sudo apt clean && sudo apt autoremove -y*     * 1. **- Download the source code of the latest stable version of the Linux kernel.**   wget -P ~/ https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.8.1.tar.xz   * 1. **- Unpack the tarball you just downloaded to your home folder.**   **tar -xvf ~/linux-5.8.1.tar.xz -C ~/**   * 1. **- Reboot your computer.**   **Section 2 - Creation**  *In this section, you will write a basic system call in C and integrate it into the new kernel.*  **2.1 - Check the version of your current kernel.**  uname -r  **2.2** **- Change your working directory to the root directory of the recently unpacked source code.**  cd ~/linux-5.8.1/  **2.3** **- Create the home directory of your system call**.  *Decide a name for your system call, and keep it consistent from this point onwards. I have chosen identity.*  mkdir identity  **2.4** **- Create a C file for your system call.**  *Create the C file with the following command.*  nano identity/identity.c    *Write the following code in it.*  #include <linux/kernel.h>  #include <linux/syscalls.h>  SYSCALL\_DEFINE0(identity){  printk("I am Jihan Jasper Al-rashid.\n");  return 0;  }  *You can write anything you like here.*  *Save it and exit the text editor.*    ***2.5 -*** **Create a Makefile for your system call.**  *Create the Makefile with the following command.*  *nano identity/Makefile*  *Write the following code in it.*  *obj-y := identity.o*  *Save it and exit the text editor.*  ***2.6 -*****Add the home directory of your system call to the main Makefile of the kernel.**  *Open the Makefile with the following command.*  *nano Makefile*  *Search for core-y. In the second result, you will see a series of directories.*  *kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/*  *In the fresh source code of Linux 5.8.1 kernel, it should be in line 1073.*  *Add the home directory of your system call at the end like the following.*  *kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/ identity/*  *Save it and exit the editor.*    **2.7** **- Add a corresponding function prototype for your system call to the header file of system calls.**  *Open the header file with the following command*.  nano include/linux/syscalls.h  *Navigate to the bottom of it and write the following code just above #endif.*  asmlinkage long sys\_identity(void);  *Save it and exit the editor.*    **2.8****- Add your system call to the kernel's system call table.**  *Open the table with the following command.*  *nano arch/x86/entry/syscalls/syscall\_64.tbl*  *Navigate to the bottom of it. You will find a series of x32 system calls. Scroll to the section above it. This is the section of your interest. Add the following code at the end of this section respecting the chronology of the row as well as the format of the column. Use Tab for space.*  *440 common identity sys\_identity*  *In the fresh source code of Linux 5.8.1 kernel, the number for your system call should be 440.*  *Save it and exit the editor.* | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | | | |  |
|  | **Section 3 – Installation**  *In this section, you will install the new kernel and prepare your operating system to boot into it.*  **3.1 - Configure the kernel.**  Make sure the window of your terminal is maximized.  Open the configuration window with the following command.  make menuconfig  Use **Tab** to move between options. Make no changes to keep it in default settings.  Save and exit.    **3.2** - **Find out how many logical cores you have.**  nproc  *The following few commands require a long time to be executed. Parallel processing will greatly speed them up. For me, it is 4. Therefore, I will put 4 after -j in the following commands.*  **3.3** - **Compile the kernel's source code**.  make -j4  **3.4** **- Prepare the installer of the kernel.**  sudo make modules\_install -j4  **3.5** - **Install the kernel**.  sudo make install -j4  **3.6** - **Update the bootloader of the operating system with the new kernel.**  sudo update-grub  **3.7** - **Reboot your computer.**      **Section 4 - Result**  *In this section, you will write a C program to check whether your system call works or not. After that, you will see your system call in action.*  **4.1** **- Check the version of your current kernel.**  uname -r  **4.2** **- Change your working directory to your home directory.**  cd ~  **4.3** - **Create a C file to generate a report of the success or failure of your system call**.  *Create the C file with the following command*.  nano report.c  Write the following code in it.  #include <linux/kernel.h>  #include <sys/syscall.h>  #include <stdio.h>  #include <unistd.h>  #include <string.h>  #include <errno.h>  #define \_\_NR\_identity 440  long identity\_syscall(void)  {  return syscall(\_\_NR\_identity);  }  int main(int argc, char \*argv[])  {  long activity;  activity = identity\_syscall();  if(activity < 0)  {  perror("Sorry, Jasper. Your system call appears to have failed.");  }  else  {  printf("Congratulations, Jasper! Your system call is functional. Run the command dmesg in the terminal and find out!\n");  }  return 0;  }  *You can customize the messages for failure and success anyhow you like.*  *Save it and exit the editor.*  **4.4** - **Compile the C file you just created.**  gcc -o report report.c  **4.5** **- Run the C file you just compiled**.  ./report  *If it displays the following, everything is working as intended.*  *Congratulations, Jasper! Your system call is functional. Run the command dmesg in the terminal and find out!*    **References**   * <https://dev.to/jasper/adding-a-system-call-to-the-linux-kernel-5-8-1-in-ubuntu-20-04-lts-2ga8> * <https://medium.com/anubhav-shrimal/adding-a-hello-world-system-call-to-linux-kernel-dad32875872> * <https://www.kernel.org/doc/html/latest/process/adding-syscalls.html> | | | |  |
|  | |  |  |